The 3DVDM Approach: A Case Study with Clickstream Data

Böhlen, Michael Hanspeter ; Bukauskas, Linas ; Mazeika, Arturas ; Mylov, Peer

Abstract: Clickstreams are among the most popular data sources because Web servers automatically record each action and the Web log entries promise to add up to a comprehensive description of behaviors of users. Clickstreams, however, are large and raise a number of unique challenges with respect to visual data mining. At the technical level the huge amount of data requires scalable solutions and limits the presentation to summary and model data. Equally challenging is the interpretation of the data at the conceptual level. Many analysis tools are able to produce different types of statistical charts. However, the step from statistical charts to comprehensive information about customer behavior is still largely unresolved. We propose a density surface based analysis of 3D data that uses state-of-the-art interaction techniques to explore the data at various granularities.

DOI: https://doi.org/10.1007/978-3-540-71080-6_2

The 3DVDM Approach
A Case Study with Clickstream Data

Michael Böhlen Linas Bukauskas Arturas Mazeika Peer Mylov

[1]Department of Computer Science, Fredrik Bajers Vej 7E
[2]Institute of Communication, Niels Jernes Vej 12

Aalborg University, 9220 Aalborg Øst, Denmark

Abstract. Clickstreams are among the most popular data sources because Web servers automatically record each action and the Web log entries promise to add up to a comprehensive description of behaviors of users. Clickstreams, however, are large and raise a number of unique challenges with respect to visual data mining. At the technical level the huge amount of data requires scalable solutions and limits the presentation to summary and model data. Equally challenging is the interpretation of the data at the conceptual level. Many analysis tools are able to produce different types of statistical charts. However, the step from statistical charts to comprehensive information about customer behavior is still largely unresolved. We propose a density surface based analysis of 3D data that uses state-of-the-art interaction techniques to explore the data at various granularities.

1 Introduction

Visual data mining is a young and emerging discipline that combines knowledge and techniques form a number of areas. The ultimate goal of visual data mining is to devise visualizations of large amounts of data that *facilitate the interpretation* of the data. Thus, visual data mining tools should be expected to provide informative but not necessarily nice visualizations. Similarly, visual data mining tools will usually enrich or replace the raw data with model information to support the interpretation. These goals, although widely acknowledged, often get sidelined and are dominated by the development of new visual data mining tools. Clearly, data analysis tools are important but it is at least as important to design principles and techniques that are independent of any specific tool.

We present an interdisciplinary approach towards visual data mining that combines mature and independent expertise from multiple areas: database systems, statistical analysis, perceptual psychology, and scientific visualization. Clearly, each of these areas is an important player in the visual data mining process. However, in isolation each area also lacks some of the required expertise. To illustrate this, we briefly recall three very basic properties that any realistic visual data mining system must fulfill. Typically each property is inherent to one area but poorly handled in the other areas.

Relevant data At no point in time is it feasible to load or visualize a significant part, let alone all, of the available data. The data retrieval must be limited to the relevant

part of the data. The relevant part of the data might have to depend on the state of the data mining process, e.g., be defined as the part of the data that is analyzed at a specific point in time.

Model information It is necessary to visualize model information rather than raw data. Raw data suffers from a number of drawbacks and a direct visualization is not only slow but makes the quality depend on the properties of the data. A proper model abstracts from individual observations and facilitates the interpretation.

Visual properties It is very easy to overload visualizations and make them difficult to interpret. The effective use of visual properties (size, color, shape, orientation, etc.) is a challenge [16]. Often, a small number of carefully selected visual properties tuns out to be more informative than visual properties galore.

This paper describes a general-purpose interactive visual data mining system that lives up to these properties. Specifically, we explain *incremental observer relative data extraction* to retrieve a controlled superset of the data that an observer can see. We motivate the use of *density information* as an accurate model of the data. We visualize *density surfaces* and suggest a small number of interaction techniques to explore the data. The key techniques are animation, conditional analyzes, equalization, and windowing. We feel that these techniques should be part of any state-of-the-art visual data mining system.

Animation Animation is essential to explore the data at different levels of granularity and get an overview and feel for the data. Without animations it is very easy to draw wrong conclusions.

Windowing For analysis purposes it is often useful to have a general windowing functionality that allows to selectively collapse and expands individual dimensions and that can be used to change the ordering of (categorical) attribute values.

Equalization The data distribution is often highly skewed. This means that visualizations are dominated by very pronounced patterns. Often these patterns are well known (and thus less interesting) and they make it difficult to explore other parts of the data if no equalization is provided.

Conditional Analyzes Data sets combine different and often independent information. It must be possible to easily perform and relate analyzes on different parts of the data (i.e., conditional data analyzes).

In Section 4 we will revisit these techniques in detail and discuss how they support the interpretation of the data. Throughout the paper we use clickstream data to illustrate these techniques and show how they contribute to a robust interpretation of the data. Clickstream data is a useful yardstick because it is abound and because the amount of data requires scalable solutions. We experienced that a number of analysis tools exhibit significant problems when ran on clickstream data, and the performance quickly became prohibitive for interactive use.

There are a number of tools that offer advanced visual data mining functionality: GGobi [3], MineSet [4], Clementine [1], QUEST [5], and Enterprise Miner [2]. Typically, these are comprehensive systems with a broad range of functionalities. Many of them support most of the standard algorithms known from data mining, machine learning, and statistics, such as association rules, clustering (hierarchical, density-based, Kohonen, k-means), classification, decision trees, regression, and principal components.

None of them supports data mining based density surfaces for 3D data and advanced interaction techniques for the data analyst. Many of the systems also evolved from more specialized systems targeted at a specific area.

Data access structures are used to identify the relevant data and are often based on the B- and R-tree [9]. R-tree based structures use minimum bounding rectangles to hierarchically group objects and they support fast lookups of objects that overlap a query point/window. Another family of access structures is based only on space partitioning as used in the kd-tree [17]. Common to all these structures is a spatial grouping. In our case the visible objects are not necessarily located in the same area. Each object has its own visibility range and therefore the visible objects may be located anywhere in the universe. In addition to the lack of a spatial grouping of visible objects the above mentioned access structures also do not support the incremental extraction of visible objects, which we use to generate the stream of visible objects.

For the model construction we estimate the probability density function (PDF). Probability density functions and kernel estimation have been used for several decades in statistics [18, 19, 7, 6, 23]. The main focus has been the precision of the estimation, while we use it to interactively compute density surfaces. We use the density surfaces to partition the space. Space partitioning has been investigated in connection with clustering [10, 8, 22, 24, 11, 20] and visualization. The visualization is usually limited to drawing a simple shape (dot, icon glyph, etc.) for each point in a cluster [11, 20] or drawing (a combination of) ellipsoids around each cluster [21, 15]. The techniques are usually not prepared to handle data that varies significantly in size, contains noise, or includes multiple not equally pronounced structures [14]. Moreover, a different density levels often requires a re-computation of the clusters, which is unsuitable for interactive scenarios let alone animation.

Structure of the paper: Section 2 discusses clickstreams. We discuss the format of Web logs, show how to model clickstreams in a data warehouse, and briefly point to current techniques for analyzing clickstreams. Section 3 discusses selected aspects of the 3DVDM System, namely the use of incremental observer relative data extraction to generate a stream of visible observations, and the model computation from this stream. Section 4 uses the clickstream data to illustrate data mining based on density surfaces.

2 Clickstreams

One of the most exciting data sources are Web logs (or clickstreams). A clickstream contains a record for every page request from every visitor to a specific site. Thus, a clickstream records every gesture each visitor makes and these gestures have the potential to add up to comprehensive descriptions of the behavior of users. We expect that clickstreams will identify successful and unsuccessful sessions on our sites, determine happy and frustrated visitors, and reveal the parts of our Web sites are (in)effective at attracting and retaining visitors.

2.1 Web Server Log Files

We start out by looking at a single standardized entry in a log file of a web server:

```
ask.cs.auc.dk  [13/Aug/2002:11:49:24 +0200]
  "GET /general/reservation/cgi-bin/res.cgi HTTP/1.0"
  200  4161  "http://www.cs.auc.dk/general_info/"
  "Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)"
```

Such an entry contains the following information.

1. The IP address of the visitor. Possibly a cookie ID is included as well.
2. The date and time (GMT) of the page request.
3. The precise HTTP request. The type of request is usually `Get` or `Submit`.
4. The returned HTTP status code. For example, code 200 for a successful request, code 404 for a non-existing URL, code 408 for a timeout, etc.
5. The number of bytes that have been transferred to the requesting site.
6. The most recent referring URL (this information is extracted from the referrer field of the HTTP request header).
7. The requesting software; usually a browser like Internet Explorer or Netscape, but it can also be a robot of a search engine.

Web logs do not only contain entries for pages that were explicitly requested. It is common for a page to include links to, e.g., pictures. The browser usually downloads these parts of a document as well and each such download is recorded in the Web log. Consider the two example entries below. The first entry reports the download of the root page. The second entry is the result of downloading an image that the root page refers to.

```
0x50c406b3.abnxx3.adsl-dhcp.tele.dk
  [13/Aug/2002:11:49:27 +0200]
  "GET / HTTP/1.1"
  200  3464  "-"
  "Mozilla/4.0 (compatible; MSIE 5.0; Windows NT; DigExt)"

0x50c406b3.abnxx3.adsl-dhcp.tele.dk
  [13/Aug/2002:11:49:27 +0200]
  "GET /images/educate.jpg HTTP/1.1"
  200  9262  "http://www.cs.auc.dk/"
  "Mozilla/4.0 (compatible; MSIE 5.0; Windows NT; DigExt)"
```

2.2 Modeling Clickstreams in a Data Warehouse

We are a long way from inferring user behavior just by analyzing the entries in a web log. We have to clean the raw log data and organize it in a way that supports the business perspective. A well known approach is to use ETL (extract transform load) techniques to pre-process and load the data into a data warehouse that supports business analyzes. Figure 1 shows an example data warehouse schema.

The schema in Figure 1 is a classical data warehouse start schema [12]. The Clickstream fact table contains an entry for every record in the Web log and is several GB large. Each entry has a number of key attributes that point to descriptive entries in the

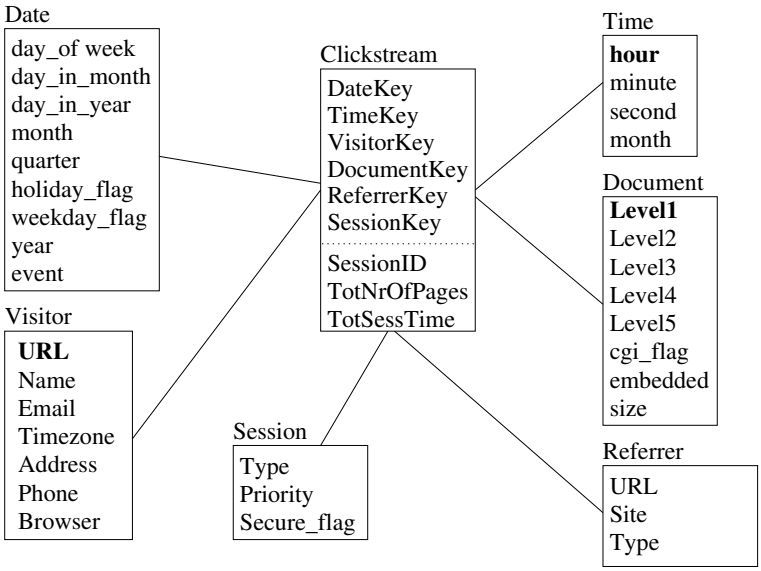


Fig. 1. Star Schema of a Clickstream Data Warehouse

dimension tables. The measure attributes carry information about individual sessions (ID, total number of pages accessed during a session, total session time). Note that these attributes are not available in the log file. They have to be inferred during the data preprocessing phase. A common approach is to sort the log on IP and time, and then use a sequential scan to identify sequences of clicks that form a session. Below we briefly describe those aspects of the dimensions that have unique properties related to the analysis of clickstreams [13].

Date Dimension The date dimension is small and has a few thousand records at most (365 days per year). The business requirements determine the attributes that should be included. For example, we should record for each day whether it is a holiday or not if some of our analyzes will treat holidays separately.

Time Dimension All times must be expressed relative to a single standard time zone such as GMT that does not vary with daylight savings time. The time dimension has 86,400 records, one for each second of a given day. A separate time dimension allows analyzes across days and makes it easy to constrain the time independent of the day.

Visitor Dimension It is possible to distinguish three types of visitors. Anonymous visitors are identified by their IP addresses. In many cases, the IP address only identifies a port on the visitor's Internet service provider. Often, these ports are dynamically reassigned, which makes it difficult to track visitors within a session let alone across sessions. A cookie visitor is one who has agreed to store a cookie. This cookie is a reliable identifier for a visitor machine. With a cookie, we can be pretty sure that a given machine is responsible for a session, and we can determine when the machine will visit us

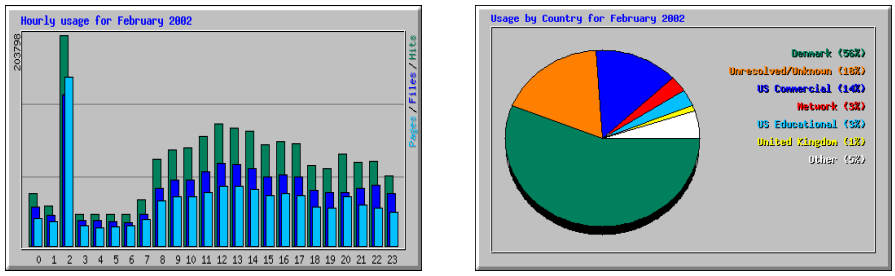
again. An authenticated visitor is the human-identified visitor who not only has accepted our cookie but sometime in the past has revealed the name and other information. The visitor dimension is potentially huge but its size can often be reduced significantly. For example, anonymous visitors can be grouped according to domain (and sub-domain). For cookie and authenticated visitors it is likely that we want to build up individual profiles, though.

Document Dimension The document dimension requires a lot of work to make the clickstream source useful. It is necessary to describe a document by more than its location in the file system. In some cases, the path name to the file is moderately descriptive (e.g., a .jpg extension identifies a picture), but this is certainly not always the case. Ideally, any file on a Web server should be associated with a number of categorical attributes that describe and classify the document.

Session Dimension The session dimension is a high-level diagnosis of sessions. Possible types are student, prospective student, researcher, surfer, happy surfer, unhappy surfer, crawler, etc. The session information is not explicitly stored in the Web log file. Basically, it has to be reverse engineered from the log file and added during data preprocessing.

2.3 Analyzing Clickstreams

There are a number of standard statistical tools available that are being used to analyze web logs. These tools are used to summarize the data and display the summaries. Two typical charts are shown in Figure 2.



(a) Histogram (by Hour)

(b) Pie Chart (by Country)

Fig. 2. Statistical Analysis of a Web Log

Typically, the summaries are computed for one dimension as illustrated by the charts in Figure 7(a). 2D (or even 3D) summaries are more rare, although some of the tools offer (2D) crosstabing. As a result the information is often spread over a large number of tables and graphs, and it is difficult to analyze a clickstream comprehensively.

We propose to combine much of this information into a single graph and use state-of-the-art interaction techniques to analyze the data from different perspectives and at different granularities.

3 The 3DVDM System

3.1 Overall Architecture

As mentioned in the introduction visual data mining is fundamentally inter-disciplinary. Any visual data mining system should make this a prime issue for the basic design of the system. An obvious approach is to go for a modular system with independent components as illustrated in Figure 3.

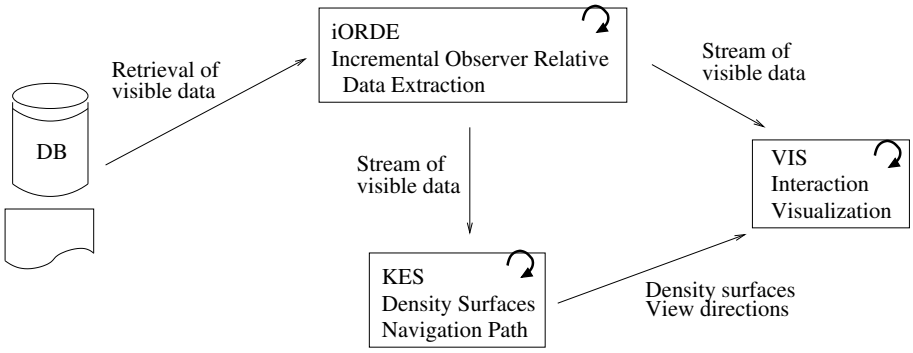


Fig. 3. 3DVDM System Architecture

The modules are implemented as separate processes that communicate through streams. Each module has an event handler that reacts to specific events. These events can be triggered by any of the other modules. Other than this no cooperation between or knowledge of the modules is required, which is an asset for a true interdisciplinary system.

The 3DVDM System is being used in environments (Panorama, Cave) where the data analyst has the possibility to explore the data from the inside and outside. BElow we describe techniques to retrieve and stream the data during such explorations and show how to process the stream and compute model information.

3.2 Streaming Visible Data

Visible objects are identified by the distance in which the object is seen. We use the *visibility range* to define the visibility of an object. In the simplest case it is proportional to the size of an object: $VR(o_i) = o_i[s] \cdot c$. Here, $VR(o_i)$ is the visibility range of object o_i, $o_i[s]$ is the size of the object, and c is a constant. Thus, object o_i will be visible in the circular or hyper-spherical area of size $VR(o_i)$ around the center of o_i.

We write $\mathcal{V}_{P_l}$ to denote the set of visible objects from the point P_l. Objects that become visible when an observer moves from position P_l to position P_{l+1} are denoted by $\Delta^+_{P_l,P_{l+1}}$. With this the stream of visible data from position P_0 to P_k is defined as a sequence $\mathcal{S}(P_0, P_k) = \langle \mathcal{V}_{P_0}, \Delta^+_{P_0,P_1} \ldots \Delta^+_{P_{k-2},P_{k-1}} \rangle$. Here, k is a number of observer positions and the number of stream slices. The definition says that we stream all visible data for the initial observer position. Any subsequent slices are produced as increments.

Figure 3.2 shows a part of the clicktream data. Specifically, the clicks from four domains are shown: .it, .de, .uk, and .es. We assume equally sized visibility ranges for all clicks (alternatively, the VR could be chosen to be proportional to, e.g., the size of the downloaded document) and let the observer move along the clicks from .it.

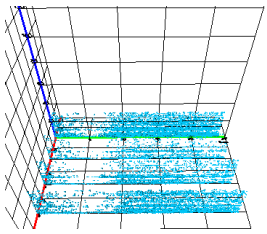
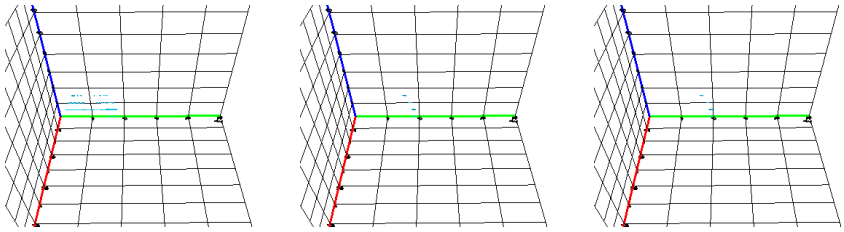


Fig. 4. The Universe of Data

The observer starts at the time position 5 and the VR is 4. Initially, 1728 visible object are received. In Figure 5(b) and 5(c) the incremental slices are shown. $\Delta^+_{P_0,P_1}$ contains 503 and $\Delta^+_{P_1,P_2}$ 363 newly visible objects.

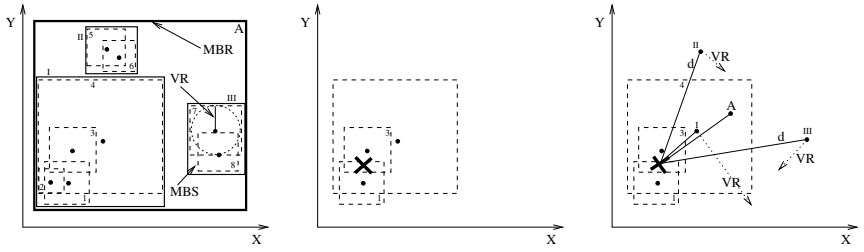


(a) Visible Objects (b) $\Delta^+_{P_0,P_1}$ (c) $\Delta^+_{P_1,P_2}$

Fig. 5. Stream of Visible Data

To provide efficient streaming of visible objects we use the VR-tree, an extension of the R-tree, together with the Volatile Access Structure (VAST). The structures enable a fast and scalable computation of Δ^+ when the observer is moving.

VAST is called *volatile* because it is initialized and created when the observer starts moving. It is destroyed if the observer no longer needs streaming support. Figure 6 shows an example of VAST that mirrors a subset of the VR-tree. Figure 6(a) shows the hierarchical structure of the VR-Tree. Figure 6(b) shows only the visible objects. These are all objects with a VR that includes the observer position. VAST mirrors the visible part of the VR tree as illustrated in Figure 6(c). In contrast to the VR tree, which uses absolute Cartesian coordinates, VAST uses an observer relative representation with distances and angles. This representation is more compact and supports a moving observer.



(a) VR-Tree (b) Visible Objects (c) VAST

Fig. 6. The VR-Tree migration.

3.3 Model Computation

The visible data is streamed to the model computation module where it is processed in slices to estimate the PDF. The estimation algorithm starts with a coarse uniform grid data structure and processes the first slice of the data. The estimation process can be in two states: the estimation or skipping state. In the estimation state the algorithm refines the PDF by adding new kernels and refining the grid structure in regions where the PDF is non-linear. The process continues until the desired precision is reached. When the estimation is precise enough it enters the skipping state. In the skipping state the algorithm skips the slices until it finds new information that is not yet reflected in the PDF. If such a slice is found the processing is resumed. The individual steps of the stream-base processing are shown in code fragment 1.

Code Fragment 1 *Estimation of the Probability Density Function*

```
Algorithm: estimatePDF
Input:
  vDataset: sequence of data points
  ε: accepted precision
  InitG: initial number of grids

Output:
  APDF tree a
```

```
Body:
  1. Initialize a
  2. skipState = FALSE
  3. FOR EACH slice s_i DO
       3.1 IF !skipState THEN
             3.1.1 Process slice s_i.
             3.1.2 Split the space according to the precision of the estimation
             3.1.3 IF precisionIsOK(a) THEN
                     skipState = TRUE
                   END IF
       3.2 ELSIF newStream(s_i) THEN
             3.2.1 skipState = FALSE
           END IF
     END FOR
```

The density surface module can be in one of two states: active (in the process mode) and inactive (waits for an wake up event). The module is active if: (i) a change of the input parameters has triggered the recalculation of the PDFs and/or density surfaces or (ii) the animation of density surfaces is switched on.

Pseudo Code Fragment 2 sketches the event handling. The first block handles new data. A new (or changed) data set triggers the re-computation of the model. The block deletes previous model information if requested (sub-block 1.1), splits the dataset into conditional datasets, and calculates the PDFs for the individual datasets (sub-block 1.3). The second block handles animation events. It calculates the next density surface level (sub-blocks 2.1-2.2) and computes the corresponding density surfaces (sub-blocks 2.3-2.4). The third block handles events triggered by a change of the density level. This requires a recalculation of the density surfaces at the requested density level. The 4th block visualizes the computed density surfaces and, optionally, corresponding data samples.

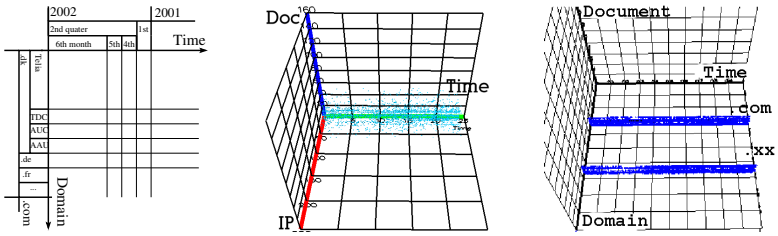
Code Fragment 2 *Dispatcher of the DS module*

```
1. IF new(vDataset) THEN
   1.1 IF reset THEN
         vPDF = vSurfaces = vPlacement = ∅
       END IF
   1.2 IF bConditional THEN
         Split vDataset according to the categorical attribute into D_1,...,D_k.
       ELSE
         D_1 = vDataset
       END IF
   1.3 FOR EACH Dataset D_i DO
         vPDF = vPDF ∪ estimatePDF(D_i, EstErr, InitG)
         vPlacement = vPlacement ∪ cPlacement
       END FOR
   END IF

2. IF bAnimate THEN
      2.1 animateDSLevel += 1.0 / animateDSFrames
      2.2 IF animateDSLevel >= 1.0 THEN
            animateDSLevel = 1.0 / animateDSFrames
      END IF
      2.3 vSurface = ∅
      2.4 FOR EACH vPDF_i DO
          vSurface = vSurface ∪
          calculateDS(vPDF_i, animateDSLevel, iDSGridSize, bEqualized)
          END FOR
      2.5 Request for calculation
    END IF
```

```
3. IF !bAnimate AND changed(fDSLevel) THEN
      3.1 FOR EACH vPDFᵢ DO
          vSurface = vSurface ∪
           calculateDS(vPDFᵢ, fDSLevel, iDSGridSize, bEqualized)
   END IF

4. visualizeObs()
```

4 Data Analyzes and Interpretation

This section discusses and illustrates the four key interaction techniques: animation, conditional analyzes, equalization, and windowing. These are general techniques that support the interpretation of the data and are useful in other settings as well. Figure 7 illustrates the challenges the data mining process has to deal with. Neither the visualization of the entire data set (Figure 7(b)) nor the visualization of two domains only (Figure 7(c)) provide detailed insights. We discuss how the above techniques help to investigate and interpret this data.



(a) Warehouse (b) All data (c) .dk and .com

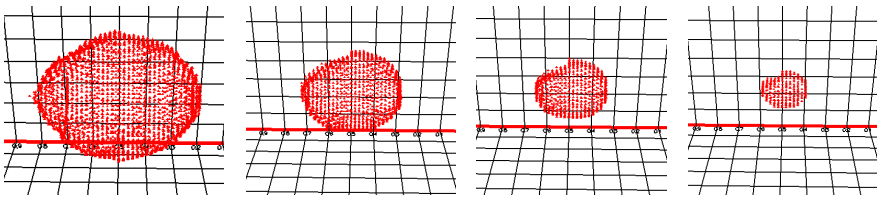
Fig. 7. Direct Visualization of a Clickstream Data Warehouse

Below we use I for the interpretation of a visualization. We use I to relate different visualization techniques. Specifically, we formulate theorems that relate the information content of different density surface visualizations.

4.1 Animation

An animation of density surfaces is a (cyclic) visualization of a sequence of density surfaces with a decreasing density level. For example if the number of frames is chosen to be four then the density surfaces at levels $\alpha_1 = 1/5$, $\alpha_2 = 2/5$, $\alpha_3 = 3/5$, and $\alpha_4 = 4/5$ will be visualized in sequence. Since the printed version of the manual is limited and cannot show the animation we present the animation of density surfaces by a sequence of snapshots at different density levels. Figure 8 shows an animated solid sphere (a 3D normal distributions)[1].

[1] Because of rescaling the sphere is deformed and resembles an ellipsoid.



(a) $\alpha = 1/5$ (b) $\alpha = 2/5$ (c) $\alpha = 3/5$ (d) $\alpha = 4/5$

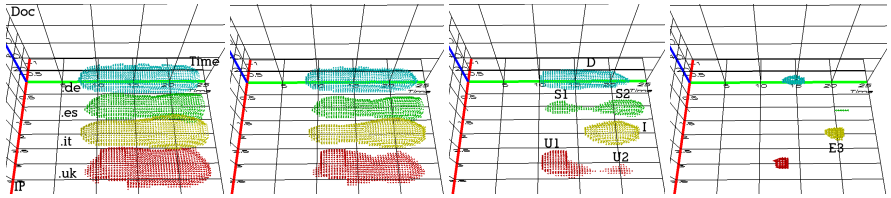
Fig. 8. Density Surfaces for Different Surface Grid Size g_s

Theorem 1. *Animated density surfaces are strictly more informative than any individual density surface:*

$$\forall k(I[\cup_i DS(D, \alpha_i)] \sqsupset I[DS(D, \alpha_k)])$$

Figure 8, which exhibits a very simple and regular data distribution, illustrates Theorem 1 quite nicely. It is necessary to look at the animated density surfaces to confirm the normal distribution. None of the four snapshots would allow us to infer the same interpretation as the four snapshots together. Note that for any given density level α_k it is possible to construct a dataset D', such that $I[DS(D, \alpha_k)] = I[DS(D', \alpha_k)]$ and $\forall i \neq k : I[DS(D, \alpha_i)] \neq I[DS(D', \alpha_i)]$.

Figure 9 shows animated density surfaces for a subset of the clickstream data set. The figure shows the clicks from the following domains: Italy (30%), Germany (28%), the UK (22%), and Spain (20%).



(a) $\alpha = 0.2$ (b) $\alpha = 0.3$ (c) $\alpha = 0.5$ (d) $\alpha = 0.9$

Fig. 9. .it, .de, .uk, .es

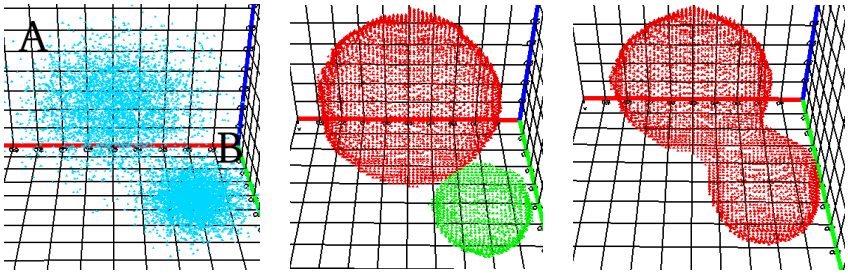
The four countries are very different in culture and working style. The density surfaces nicely reflect this. For example in Spain people have a long break (siesta) in the middle of the day. The siesta divides the density surface for Spain into two similar sized surfaces (cf. $S1$ and $S2$, Figure 9(c)). Italy produces most of the clicks during the second part of the day (cf. surface I, Figure 9(c)). In contrast most of the UK clicks are

received in the first part of the day (cf. surface $U1$, Figure 9(c)). The German domain is bound by the working hours and peaks in the afternoon. Figure 9(d) shows the peaks for the individual domains.

4.2 Conditional Density Surfaces

Conditional density surfaces are the result of splitting a data set into several subsets and computing a model (i.e., density surfaces) for each of them. A common approach is to split a data set according to the values of a categorical attribute (e.g., sex, country, etc).

An example of conditional density surfaces is shown in Figure 10. The data set contains two normal distributions. A binary attribute W was added to assign the obser-



(a) The Dataset (b) Conditional (c) Unconditional

Fig. 10. Difference between Conditional and Unconditional Analysis

vations to the clusters. All observations that fall into the cluster A have $W = 1$, while observations that fall into cluster B have $W = 2$. Figures 10(c) shows a non-conditional density surface. A single density surface encloses both structures in space. Figure 10(b) shows conditional density surfaces. The conditional analysis separates the data points and yields two independent (but possibly intersecting) density surfaces.

Theorem 2. *Let $D_1, ..., D_k$ be independent data sets with identical schemas $\mathbf{D_i}$, and let $D = \cup_{i=1}^{k} \pi[\mathbf{D_i}, i](D_i)$ be the union of these data sets with schema $\mathbf{D} = \mathbf{D_i} \cup \{W\}$. If a conditional data set contains independent structures then an appropriate conditional analysis is more informative than an unconditional analysis.*

$$I[\cup_i DS(\sigma[W = i](D), \alpha)] \sqsupseteq I[DS(D, \alpha)])$$

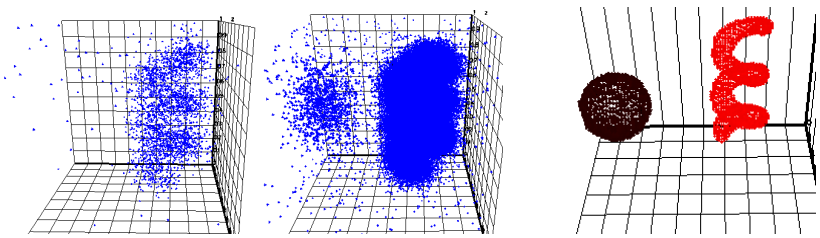
Basically, a conditional analysis will yield the same result as an (unconditional) analysis of the individual data sets. If the data sets are independent this is the optimal result. In practice, conditional analyzes can also be useful if independence is not achieved completely. For example, it can be appropriate to treat the clicks from different countries as being independent even if this is not completely accurate. Often it is

best to try out both types of analysis, which means that switching between consitional and unconditional analyzes should be supported by the interaction tools available to the data analyst.

4.3 Equalization of Structures

Many real world data sets contain very skewed data. For example, it is likely that a web server will handle a huge amount of local traffic. This does not necessarily mean that non-local domains are less important for the data analysis. In order to support the exploration of not equally pronounced structures it is necessary to equalize the structures before they are displayed.

The data set in Figure 11 contains two structures: a spiral (80% of all observations) and a sphere (20% of all observations). In Figure 11(a) we chose the number of observations that yields the best visualization of the spiral. The result is a scatter plot that does not show the sphere. In Figure 11(b) we increase the number of observations until the sphere can be identified. This yields a scatter plot that makes it difficult to identify the spiral. Clearly, it is much easier to identify the structures if equalized density surfaces are used (cf. Figure 11(c)).

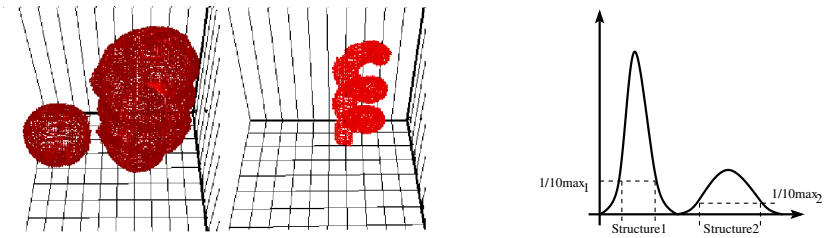


(a) $n = 3'000$ (b) $n = 100'000$ (c) Equalized DS

Fig. 11. Scatterplots versus Equalized Density Surfaces

Since the densities of the structures are very different, an overall best density level does not exist as illustrated in Figure 12(a). A low density yields an appropriate sphere but only shows a very rough contour of the spiral. A high density shows a nice spiral but does not show the sphere at all. Basically, we experience the same problems as with the scatterplot visualization. The density information makes it fairly straightforward to equalize the two structures. The idea is illustrated in Figure 12(b). The structures are equalized by adaptively (i.e., based on the density of a structure) calculating the density level for each structure (e.g., 10% of the maximum density for each structure).

Equalization requires a separation of structures. This comes for free if we use conditional density surfaces. Equalization and conditional density surfaces therefore complement each other nicely. With unconditional analyzes the separation has to be implemented as a separate process.



(a) Non-Equalized Density Surfaces (b) Lazy Equalization

Fig. 12. Equalization of Structures

Note that, in contrast to density surfaces, scatterplot visualizations are not easy to equalize. In principle, one has to determine the optimal number of observations that shall be visualized for each structure. This number depends on many factors and cannot be determined easily.

Theorem 3. *Any data set D can be completely dominated:*

$$\forall D \exists D'(I[DS(D', \alpha)] = I[DS(D \cup D', \alpha)])$$

Figure 12(a) illustrates how the spiral dominates the sphere. Here the domination is not complete because there is still a density level at which the sphere can be identified. If the density of the spiral was increased further the sphere will eventually be treated as noise.

Theorem 4. *Equalization preserves the interpretation of individual structures:*

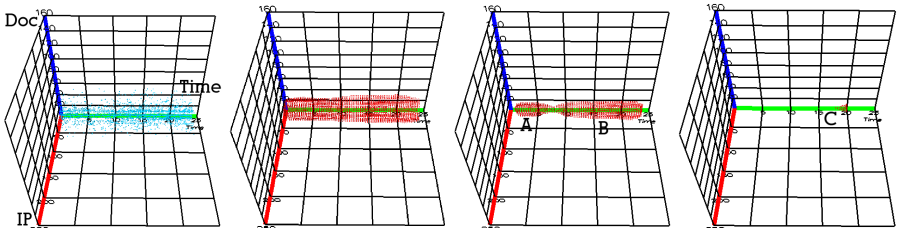
$$\forall D, D'(I[DS^{eq}(D \cup D', \alpha)] = I[DS(D, \alpha)] \cup I[DS(D', \alpha)])$$

The preservation is illustrated in Figure 11(c). Here the two structures are shown as if each of them was analyzed individually.

4.4 Windowing

The window functionality is used if we want to investigate a subset of the data in more detail. Technically, the window function filters out the observations that fall outside the selected window and re-scales the data to the unit cube. Conceptually, the window functionality enables an investigation at the micro level (a “smart” zoom into the dataset). In contrast to standard zooming techniques, windowing allows to zoom in on a single dimension. For example, it is possible to restrict the analysis to a few domains but preserve the complete time dimension. Note that windowing must trigger a recalculation of the model. This makes it possible that a single surface at the macro level can split into several surfaces at micro level, as one expects it to be.

Figure 13 is a direct visualization of the clickstream and illustrates the problem. Because the coding preserves the natural ordering of the data all frequent domains (.dk, .com, etc.) are clustered at the beginning of the axis. This yields a very skewed plot that can only be used for a simple overall load analysis: The work-load is high during the very early morning hours and the local working hours (surfaces A and B in Figure 13(c)), and the work-load peaks around 8PM (surface C in Figure 13(d)).



(a) Scatterplot (b) $\alpha = 0.2$ (c) $\alpha = 0.5$ (d) $\alpha = 0.9$

Fig. 13. The Sunsite Dataset (Natural Ordering)

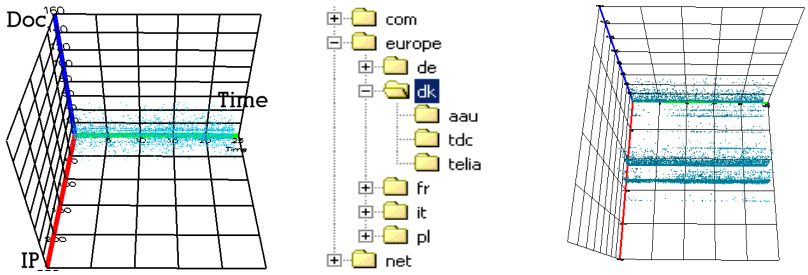
What we want is that the cluster starts to "unfold" as we move closer, i.e., we expect the domains to separate. Standard zooming techniques do not provide this functionality directly. Typically, when moving closer we not only narrow the IP domain but also loose the overview of the time domain. Moreover, zooming does not trigger a re-computation of the model, which means that the surfaces will simply be stretched.

Figure 14 illustrates the idea of the windowing functionality. A simple windowing tool is shown in Figure 14(b). The menu allows to select the domains that shall be visualized (i.e., the window). It supports dimension hierarchies and does not require that the selected domains are contiguous. The effect of zooming in is illustrated in Figure 14(c). The original cluster of points has unfold and it is now possible to analyze selected domains/regions.

Another useful feature of windowing is re-ordering. If the data analyst wants to change the ordering of the domains this can be very easily done using the tool shown in Figure 14(b).

5 Summary and Future Work

We investigated the potential of density surfaces of 3D data to analyze clickstream data. Density surfaces accurately summarize the data and yields a good model. Density surfaces are simple visual structures that are easy to perceive and support the interpretation of the data. We showed that animation, conditional analyzes, equalization, and windowing are crucial interaction techniques. They make it possible to explore the data at different granularity levels, which leads to a robust interpretation.



(a) The Scatter Plot (b) $\alpha = 0.2$ (c) $\alpha = 0.5$

Fig. 14. The Clickstream Data from www.sunsite.dk

In the future it would be interesting to classify density surfaces and come up with an alphabet. The analysis of high-dimensional data is another interesting issue. One could for example investigate projection techniques before the density surfaces are computed, or the (moderate) use of additional visual properties.

References

1. Clementine. http://www.spss.com/clementine.
2. Enterpriseminer. www.sas.com.
3. Ggobi. http://www.ggobi.org.
4. Mineset. http://www.sgi.com.
5. Quest. http://www.almaden.ibm.com/cs/quest.
6. G. C. Van den Eijkel, J. C. A. Van der Lubbe, and E. Backer. A Modulated Parzen-Windows Approach for Probability Density Estimation. In *IDA*, 1997.
7. M. Farmen and J. S. Marron. An Assesment of Finite Sample Performace of Adaptive Methods inDensity Estimation. In *Computational Statistics and Data Analysis*, 1998.
8. S. Guha, R. Rastogi, and K. Shim. CURE: an Efficient Clustering Algorithm for Large Databases. In *ACM SIGMOD International Conference on Management of Data*, pages 73–84, 1998.
9. A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In B. Yormark, editor, *SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18-21, 1984*, pages 47–57. ACM Press, 1984.
10. A. Hinneburg and D. A. Keim. Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering. In *The VLDB Journal*, pages 506–517, 1999.
11. I. Davidson and M. Ward. A Particle Visualization Framework for Clustering and Anomaly Detection. In *Proceedings of Workshop on Visual Data Mining in conjunction with SIGKDD*, 2001.
12. R. Kimball. *The Data Warehouse Toolkit*. John Wiley & Sons, Inc., 1996.
13. R. Kimball and R. Merz. *The Data Webhouse Toolkit—Building the Web-Enabled Data Warehouse*. Wiley Computer Publishing, 2000.

14. A. Mazeika, M. Böhlen, and P. Mylov. Density Surfaces for Immersive Explorative Data Analyses. In *Proceedings of Workshop on Visual Data Mining in conjunction with SIGKDD*, 2001.
15. M.H. Gross and T.C Sprenger and J. Finger. Visualizing information on a sphere. In *Visualization*, 1997.
16. H. R. Nagel, E. Granum, and P. Musaeus. Methods for Visual Mining of Data in Virtual Reality. Proceedings of the International Workshop on Visual Data Mining, in conjunction with ECML/PKDD2001 - 2th European Conference on Machine Learning (ECML'01) and 5th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'01), 2001.
17. J. T. Robinson. The K-D-B-Tree: A Search Structure For Large Multidimensional Dynamic Indexes. In Y. Edmund Lien, editor, *Proceedings of the 1981 ACM SIGMOD International Conference on Management of Data, Ann Arbor, Michigan, April 29 - May 1, 1981*, pages 10–18. ACM Press, 1981.
18. D. W. Scot. *Multivariate Density Estimation*. Wiley & Sons, New York, 1992.
19. B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986.
20. T. A. Keahey. Visualization of High-Dimensional Clusters Using Nonlinear Magnification. In *Proceedings of SPIE Visual Data Exploration and Analysis*, 1999.
21. T.C. Sprenger and R. Brunella and M.H. Gross. H-BLOB: a hierarchical visual clustering method using implicit surfaces. In *Visualization*, 2000.
22. W. Wang and J. Yang and R. R. Muntz. STING: A statistical information grid approach to spatial data mining. In *The VLDB Journal*, pages 186–195, 1997.
23. M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman & Hall, London, 1985.
24. T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: an efficient data clustering method for very large databases. In *SIGMOD Conference*, pages 103–114, 1996.